ANSWERS

F10.9 Project: reliable delivery

Networks · GCSE · OCR J277 1.3.2, AQA 8525 3.5, Edexcel 1CP2 4.1.6 · about 25 min

What this lesson is about

Deliver a report over a lossy radio with packets, acknowledgements and resending.

- The protocol
- Try one packet
- Plan it first

Questions

5 marks in all

1. What does an acknowledgement tell the sender? [1 mark]

- ☐ **A** That the packet arrived safely
- ☐ **B** That the packet was lost
- ☐ **C** The route the packet took
- ☐ **D** The receiver's IP address

> **Answer:** A. No acknowledgement means send again.

2. Which protocol resends lost packets? [1 mark]

- ☐ **A** TCP
- ☐ **B** UDP
- ☐ **C** IP
- ☐ **D** HTTP

> **Answer:** A. UDP does not check or resend.

3. Why does the sender wait for a limited time, not forever? [1 mark]

- ☐ **A** A reply may never come if the packet or the reply was lost
- ☐ **B** It saves battery
- ☐ **C** Protocols require exactly one second
- ☐ **D** To make the message shorter

> **Answer:** A. After the timeout, it sends again.

4. A message is sent in 3 packets and packet 2 has to be sent 3 times. How many transmissions is that? [1 mark]

> **Answer:** 5. 1 + 3 + 1.

5. Why would a live video call use UDP? [1 mark]

- ☐ **A** A late packet is useless, so waiting to resend it only adds delay
- ☐ **B** UDP encrypts the video
- ☐ **C** UDP guarantees every packet arrives
- ☐ **D** UDP uses less electricity

> **Answer:** A. Speed matters more than completeness for live video.

The task: deliver the report

Split `message` into packets of 14 characters, in the form `<number>/<total>:<piece>`. Deliver them to the Base with stop-and-wait. Print `sent <number>` each time you send a packet (including when you send it again) and print each reply you get. Never send the next packet before the last one is acknowledged. When every packet is acknowledged, print `delivered in <n> transmissions`, and turn the LED green.

```
# the two lines every program starts with: the commands, then the robot
from bugbot import *
connect()

message = "robot 7 reporting: mat clear, battery ok"
```

The hint students can ask for: For each packet, keep sending it until the reply is the acknowledgement for that packet's number. Count every send, including the repeats, and only move to the next packet once this one is acknowledged.

A solution

```
from bugbot import *
connect()
message = "robot 7 reporting: mat clear, battery ok"

def wait_reply(seconds=1.0):
    for tick in range(int(seconds * 10)):
        wait(0.1)
        for sender, text in messages():
            return text
    return None

pieces = [message[i:i + 14] for i in range(0, len(message), 14)]
packets = [f"{n + 1}/{len(pieces)}:{piece}" for n, piece in enumerate(pieces)]
count = 0
for n, packet in enumerate(packets, start=1):
    acked = False
    while not acked:
        send(packet)
        count = count + 1
        print("sent", n)
        reply = wait_reply()
        if reply is not None:
            print(reply)
        acked = reply == "ack " + str(n)
print("delivered in", count, "transmissions")
led(0, 255, 0)
```

Any program that meets the task's checks is marked correct in the simulator; this is one way, not the only way.